Subject: Re: [PATCH v2 09/11] memcg: propagate kmem limiting information to children
Posted by Glauber Costa on Tue, 21 Aug 2012 09:17:14 GMT
View Forum Message <> Reply to Message

On 08/21/2012 12:35 PM, Michal Hocko wrote:
> On Tue 21-08-12 09:54:30, Michal Hocko wrote:
>> E.g. how do you handle charges you left behind? Say you charged some
>> pages for stack?
>
> I got to the last patch and see how you do it. You are relying on
> free_accounted_pages directly which doesn't check kmem_accounted and
> uses PageUsed bit instead. So this is correct. I guess you are relying
> on the life cycle of the object in general so other types of objects
> should be safe as well and there shouldn't be any leaks. It is just that
> the memcg life time is not bounded now. Will think about that.
>
Unless you have a better way, I believe any kind of transversal in the
free page path is performance detrimental. So the best way is to be
explicit and mark a specific callsite as a memcg free.

As for the unbounded time, you are correct. However, I believe it is
possible to move a lot of the work we do for free (such as freeing the
percpu counters and the css_id itself) to an earlier time.

Also, if it ever becomes a problem, it is theoretically possible to
avoid this, by tracking the kmem pages in a per-memcg list. We would
then transverse such list as we do for user pages, and reparent them.
The problem is that this is also a bit space inefficient, since we can't
reuse any more fields in page_struct for the list_head, so we'd need an
external structure. There is a list_head + a pointer per tracked page.